Colaboração com Assistente de Codificação Baseado em IA: Benefícios e Desafios

Wendy Mendes, Samara Souza, Cleidson de Souza

Instituto de Ciências Exatas e Naturais - Universidade Federal do Pará - UFPA
Rua Augusto Corrêa, 1, CEP: 66075-110 - Guamá, Belém – PA – Brasil.
wendymgaleno@gmail.com, samara.souza@icen.ufpa.br, cleidson.desouza@acm.org

Abstract. *This ongoing article aims to explore the collaboration between software engineers and intelligent assistants based on (AI) in the context of software development. Adopting a research methodology that included interviews with 14 software professionals, we sought to understand their perceptions and experiences of using intelligent assistants in their work environments. The preliminary results highlight both the benefits and the challenges inherent in using these technologies. In this context, the contribution of this study lies in identifying the factors that permeate collaboration with intelligent assistants. As a result, professionals can maximize the benefits of this collaboration, provided they are aware of the associated challenges.*

Resumo. *Este artigo em andamento tem como objetivo explorar a colaboração entre engenheiros de software e assistentes inteligentes baseados em (IA) no âmbito do desenvolvimento de software. Adotando uma metodologia de pesquisa que incluiu entrevistas com 14 profissionais da área de software, buscamos compreender as percepções e experiências desses no uso de assistentes inteligentes em seus ambientes de trabalho. Os resultados preliminares destacam tanto os benefícios quanto os desafios inerentes ao emprego dessas tecnologias. Neste contexto, a contribuição deste estudo reside na identificação dos fatores que permeiam a colaboração com assistentes inteligentes. Como resultado, os profissionais podem maximizar os benefícios dessa colaboração, desde que estejam cientes dos desafios associados.*

1. Introdução

O desenvolvimento de software é impulsionado por novos desafios, tecnologias e oportunidades. Assim, os engenheiros de software estão sempre buscando maneiras de melhorar seus processos de trabalho e promover a inovação [Rajlich 2014]. Recentemente, uma tendência notável tem sido a adoção de assistentes inteligentes para geração de código. Esses assistentes tais como GitHub Copilot [GitHub 2022], ChatGPT [OpenAI 2023], Amazon CodeWhisperer [Amazon 2023], Blackbox [Blackbox 2023], alimentados por inteligência artificial (IA), têm o potencial de transformar a maneira como o software é criado [Barke et al. 2023, Bird et al. 2022, Ernst and Bavota 2022]. Nessa perspectiva, este trabalho em andamento visa explorar a seguinte questão de pesquisa: *Como engenheiros de software colaboram com os assistentes inteligentes de geração de código?*

A parceria entre engenheiros de software e assistentes inteligentes tem o potencial de trazer benefícios significativos para a indústria de desenvolvimento de software. Essa parceria promete melhorar a eficiência em diversos pontos, incluindo a

aceleração do processo de codificação [Barke et al. 2023], a tradução de código entre linguagens de programação [Weisz et al. 2022] e, de um modo geral, o aumento da produtividade [Perry et al. 2022]. Em um cenário onde a tecnologia desempenha um papel cada vez mais proeminente em diversas esferas, é imperativo reconhecer que a colaboração não se limita mais à interação humana direta. Ao invés disso, pode englobar relações simbióticas entre seres humanos e assistentes de codificação baseados em IA, proporcionando uma abordagem ampla e inovadora para a realização de tarefas complexas. No entanto, entender como essa colaboração é estabelecida e quais desafios podem surgir ao longo do caminho é essencial para maximizar os benefícios e garantir a qualidade do software resultante.

Nossa pesquisa está em andamento com o objetivo de investigar a colaboração entre engenheiros de software e assistentes inteligentes. A relevância desse estudo reside na evolução constante da tecnologia, especialmente no papel crescente desempenhado por assistentes inteligentes em ambientes de desenvolvimento de software. Ao identificar e analisar os fatores que influenciam a colaboração, tais como padrões de uso, benefícios percebidos e desafios enfrentados, podemos contribuir para o avanço do conhecimento na área, oferecendo insights importantes para os profissionais que buscam otimizar essa parceria. Esta pesquisa é relevante, pois busca não apenas explorar as potencialidades dessa colaboração, mas também conscientizar sobre os desafios, proporcionando uma base sólida para aprimoramentos futuros e avanços significativos na interação entre humanos e assistentes inteligentes na engenharia de software.

2. Referencial Teórico

O progresso das *Large Language Models* (LLM), como o GPT-3 da OpenAI [Brown et al. 2020], tem revolucionado a capacidade da IA de compreender e produzir texto de forma contextualmente relevante. Esses modelos são treinados em grandes conjuntos de dados textuais e têm a habilidade de responder perguntas, realizar traduções e criar conteúdo coerente e informativo em diferentes contextos. Este avanço tem implicações profundas em diversas áreas, desde assistentes virtuais e automação de tarefas até melhorias na escrita assistida por máquina e no desenvolvimento de software. Por exemplo, no desenvolvimento de software, esses modelos auxiliam na geração automática de código, acelerando o processo de programação e permitindo maior produtividade para os desenvolvedores.

As LLMs operam por meio de arquiteturas de redes neurais profundas, onde camadas sucessivas de neurônios artificiais processam informações textuais. Essas redes são treinadas em grandes volumes de texto para aprender padrões, estruturas gramaticais e conexões semânticas. O núcleo do funcionamento das LLMs reside na sua capacidade de prever a próxima palavra em uma sequência, dada a sequência anterior. Essa tarefa de previsão é repetida várias vezes para treinar o modelo, aprimorando sua habilidade em compreender o contexto e gerar texto coerente [Chang et al. 2023]. Com uma quantidade significativa de parâmetros ajustáveis [Barke et al. 2023], as LLMs adquirem um amplo conhecimento sobre diversos tópicos, permitindo que interpretem perguntas, criem conteúdo original e até mesmo simulem o estilo de escrita dos autores humanos. Esse mecanismo altamente adaptável para geração de texto impulsionam a capacidade das LLMs em produzir textos com alta qualidade em diversas aplicações.

Além da geração de texto, as LLMs têm desempenhado um papel crucial no desenvolvimento de código, auxiliando programadores em tarefas que variam desde as mais simples até as mais complexas. Um aspecto crítico a ser considerado nesta etapa de geração de código é a precisão dos resultados apresentados por esses assistentes virtuais. Ziegler et al. [Ziegler et al. 2022] examinaram minuciosamente a qualidade das sugestões de código geradas por LLMs e sua adequação para uso em projetos reais de desenvolvimento de software. A precisão do código gerado é uma consideração importante, pois imprecisões podem resultar em vulnerabilidades de segurança, erros de funcionamento e outros problemas que afetam diretamente a qualidade e confiabilidade do software final.

3. Trabalhos Relacionados

O estudo de [Barke et al. 2023] também se destaca como uma contribuição valiosa para a discussão sobre assistentes virtuais na geração de código. Este artigo, assim como a pesquisa de [Ziegler et al. 2022], examina a interação dos programadores com modelos geradores de código, fornecendo insights importantes sobre como os desenvolvedores colaboram com essas ferramentas. Ao analisar como os programadores aproveitam as sugestões de código oferecidas pelos assistentes, bem como as estratégias que empregam para avaliar e implementar essas sugestões de maneira precisa, o estudo "Grounded Copilot"acrescenta uma camada de compreensão fundamental sobre como a correção e eficácia das soluções de código são percebidas e integradas no processo de desenvolvimento de software. Além de analisar como esse assistente é utilizado como preenchimento automático de código e como ferramenta de exploração para quando o programador não sabe por onde começar.

Uma ferramenta amplamente usada para ajudar na criação de código é o Copilot, lançado pelo Github em junho de 2021 [Friedman 2021]. Desde que foi lançado, o Copilot tem sido integrado em várias plataformas de desenvolvimento, como o Visual Studio Code e o JetBrains. É importante destacar que o Copilot é alimentado pelos modelos Codex da OpenAI [Chen et al. 2021]. Esses modelos Codex são refinados através do ajuste fino do GPT-3 [Brown et al. 2020], usando repositórios públicos do Github como fonte de aprendizado.

Por mais que o uso de assistentes inteligentes têm crescido em grande escala nos dias atuais, é necessário atentar para os desafios enfrentados por programadores ao utilizá-los. Um dos principais desafios é a lacuna que frequentemente ocorre entre as expectativas dos programadores e suas experiências reais com essas ferramentas. Essa disparidade pode surgir devido a expectativas exageradas geradas pela publicidade e pelo entusiasmo em torno das tecnologias baseadas em grandes modelos de linguagem. Além disso, a qualidade e precisão do código gerado por esses assistentes são fundamentais, pois impactam diretamente a eficácia das ferramentas. A correção e adaptação do código gerado às necessidades específicas do projeto também representam desafios significativos [Vaithilingam et al. 2022].

Ao contrário de estudos anteriores que se concentraram na forma como os programadores usam assistentes virtuais para gerar código, nossa pesquisa vai além. Queremos entender não apenas como os desenvolvedores usam essas ferramentas, mas também como podem colaborar efetivamente com elas. Em vez de apenas apontar desafios, nossa pesquisa se compromete em encontrar estratégias práticas que melhorem a forma como

desenvolvedores e assistentes inteligentes trabalham juntos.

4. Metodologia

Nosso estudo utilizou uma abordagem qualitativa, com foco em compreender as percepções e experiências dos desenvolvedores de software, em relação ao uso de assistentes inteligentes por meio de entrevistas semiestruturadas. Nós realizamos 14 entrevistas. Em média, cada entrevista teve uma duração de 33.57 minutos (53 máx. e 18 mín.), sendo conduzidas online e gravadas com a devida autorização dos participantes. Nossa pesquisa ocorreu em duas rodadas, envolvendo diferentes fontes. Na primeira rodada, empregamos uma abordagem de mineração de dados no Twitter, com o objetivo de identificar participantes que compartilharam suas experiências em usar ferramenta de assistente inteligente (GitHub Copilot) em seu dia a dia para a escrita de código. Nessa fase, foram enviados convites para potenciais participantes, convidando-os a participar de uma entrevista, como resultado, conseguimos alcançar um total de 8 pessoas. Em seguida, na segunda etapa, realizamos entrevistas com 6 desenvolvedores de uma empresa de software estabelecida, com objetivo de aprofundar mais os dados encontrado na primeira fase. Juntas, essas duas etapas de coleta de dados proporcionaram uma visão abrangente das percepções e práticas relacionadas ao uso de assistentes inteligentes no contexto do desenvolvimento de software. Para fornecer uma visão mais detalhada dos nossos participantes, a tabela 1 apresenta informações específicas sobre cada um deles.

Os tópicos abordados em nossas entrevistas incluem: (I) coleta de dados demográficos, e (II) perguntas sobre a percepção dos profissionais sobre as assistentes inteligentes, (*qual sua percepção sobre essa ferramenta?*), os benefícios (*Que tipo de auxílio você busca na ferramenta?*), e desafios (*Existe alguma limitação que você encontrou ao utilizar a ferramenta?*). Isto nos permitiu capturar informações sobre a percepção de colaboração e aplicação prática da IA na escrita do código pelos entrevistados.

Todas as entrevistas foram transcritas e submetidas a uma análise por meio de um processo de codificação aberta, conduzida manualmente. Inicialmente, um dos autores encarregou-se de criar uma lista inicial de códigos, identificando palavras-chave ou códigos relevantes nos dados das entrevistas, e à medida que o processo avançava, a lista era revisada e refinada. Com base nessa lista inicial de códigos, realizamos análises colaborativas repetidas para identificar semelhanças entre os códigos, a fim de desenvolver códigos ou categorias de ordem superior.

Durante o processo, também reconhecemos a necessidade de aprimorar as categorias identificadas. Por fim, realizamos uma revisão minuciosa das categorias e efetuamos modificações na nomenclatura quando necessário, para garantir que representassem de maneira mais clara as informações encontradas nos dados das entrevistas. Cada transcrição foi realizada de forma colaborativa por dois autores e quaisquer conflitos foram resolvidos por meio de reuniões de discussão e consenso, um terceiro autor validou esse processo de codificação.

O objetivo dessa análise era compreender como os desenvolvedores de software realizam suas atividades de trabalho, incluindo o uso de ferramentas de IA. Além disso, buscamos identificar os benefícios e desafios específicos que esses profissionais experimentaram ao incorporar essas ferramentas em suas rotinas de trabalho.

Tabela 1. Perfil dos Entrevistados. "Exp."é usado para indicar anos de experiência e "Duração"é usado para indicar o tempo de duração da entrevista em minutos.

#	Gênero	Função	Idade.(ano)	Exp.(ano)	Uso	Duração
P1	M	Desenvolvedor	25	7	Copilot	25m
P2	M	Desenvolvedor	19	1	Copilot	22m
P3	M	Desenvolvedor	32	12	Copilot	18m
P4	M	Engenheiro de Software	23	4	Copilot/ChatGPT	34m
P5	M	Lider Técnico	32	16	Copilot	34m
P6	M	Desenvolvedor	27	2	Copilot/ChatGPT	48m
P7	M	Engenheiro de Software	30	11	Copilot	27m
P8	M	Lider Técnico	30	10	ChatGPT	26m
P9	M	Engenheiro de Software	28	7	ChatGPT	27m
P10	M	Desenvolvedor	31	7	ChatGPT	47m
P11	M	Desenvolvedor	21	1	ChatGPT	34m
P12	M	Lider de Inovação	34	16	ChatGPT	53m
P13	M	Engenheiro de Dados	28	3	ChatGPT	37m
P14	M	Engenheiro de Software	32	8	ChatGPT	38m
		Média	28	7.5	-	33.57 m
		Desvio Padrão	4.52	5.05	-	10.34 m
		Máx	34	16	-	53 m
		Mín	19	1	-	18 m

5. Resultados Parciais

5.1. Benefícios

Uma das questões fundamentais abordadas durante as entrevistas com os profissionais de software diz respeito ao momento em que eles utilizam a IA em suas atividades de desenvolvimento. Nesta seção, exploraremos os diversos benefícios percebidos pelos desenvolvedores ao introduzir ferramentas de assistência inteligente em seus fluxos de trabalho de escrita do código.

5.1.1. Refinando o Foco

Um benefício que os entrevistados destacaram ao usar assistentes inteligentes em seu fluxo de trabalho é para acelerar o processo de criação de código, o que se traduz em economia de tempo e na capacidade de concluir tarefas com mais rapidez, permitindo que eles concentrem seu tempo em pontos mais complexos e estratégicos, como citado por P5: ("*consigo notar nitidamente uma melhora na minha produtividade, sem dúvida. Ainda que seja, como eu falei, coisas triviais ou definir um looping, eu vou fazer isso daí em alguns segundos. Mas eu não quero fazer, eu quero que ela (IA) complete ali rapidamente para eu pensar no problema que eu tenho para fazer de verdade.").* Outro entrevistado acrescenta, ("*acho que acelera, a gente deixa de fazer coisas mais operacionais, e pensa mais em coisas de inteligência, então acho que o trabalho vai evoluir para uma de [. . .]*

eu vou pensar mais no negócio e menos no jeito que eu vou implementar. Esses tipos de ferramentas ajudam a gente pensar mais em negócio e menos nas coisas repetitivas que todo negócio tem que ter"), P3.

Além disso, o entrevistado P11 relata que pensa nessa ferramenta como uma camada de abstração, ("*Na minha percepção, eu vejo ele como uma nova camada de abstração de código. Do mesmo jeito que a gente não precisa escrever em assembly, pois existem linguagens de mais alto nível, para mim é a mesma coisa, é uma camada a mais, eu vejo que em alguns anos estaremos escrevendo mais regras de negócio e menos implementação").*

Isso sugere que essa colaboração pode promover uma melhor alocação de tempo e recursos, direcionando o foco dos programadores para atividades de maior relevância e complexidade. Esses benefícios evidenciam o potencial dessa tecnologia em aprimorar o desenvolvimento de software e otimizar a produtividade dos profissionais.

5.1.2. Ponto de Partida

Enquanto muitos o utilizam para complementar trechos de código com os quais já possuem experiência na linguagem de programação, muitos outros recorrem à IA como um ponto de partida para iniciar tarefas, no caso, eles usam para ajudá-los a começar uma tarefa que não tem conhecimento da estrutura ou APIs. Como ilustrado por P1: ("*Então já teve vezes que eu tinha que fazer uma função que executava determinada ação e eu não sabia como exatamente teria que ser, eu só sabia mais ou menos por cima e a ferramenta foi lá e já me deu tudo na mão")*, o participante P2 também compartilha: ("*Mas se acabar de eu não conhecer direito a API que eu estou trabalhando de forma bem conhecida, mas eu não tenho muita familiaridade com ela. Eu, às vezes, deixo ele fazer e me dizer como é que aquilo a princípio é implementado").*

O parcipante P7 relata que ("*Quando eu não tenho muita ideia do que fazer, eu acho legal ter a sugestão da Copilot pois pelo menos eu tenho como um start point").* Este insight do P7 enfatiza o papel interessante que o Copilot desempenha para os desenvolvedores quando eles encontram incertezas ou não têm uma direção clara em suas tarefas de codificação. O Copilot, neste contexto, serve como um recurso benéfico, fornecendo um ponto fundamental para os desenvolvedores iniciarem seu trabalho e superarem desafios relacionados à idealização e início de tarefas. Além disso, é importante observar que o ChatGPT também contribui para esse aspecto, o P11 afirma: ("*Por exemplo, quando eu quero fazer otimização de um código, ou tirar uma dúvida. Aí o ChatGPT consegue me dar uma solução alternativa para o que eu quero fazer").*

Podemos concluir que os relatos dos entrevistados demonstram que essa abordagem oferece uma orientação inicial, pois ao pedi auxílio da IA e ela fornece insights iniciais, os desenvolvedores podem começar suas tarefas mesmo quando estão trabalhando com novas linguagem e API's desconhecidas ou com pouca experiência. Esses benefícios destacam o potencial dessa tecnologia para aprimorar o desenvolvimento de software, aumentando significativamente a produtividade dos profissionais envolvidos

5.1.3. Otimização de Tarefas

Observamos que os desenvolvedores adotam a ferramenta de IA como um recurso de suporte para complementar tarefas específicas de criação de código. Como evidenciado por P9, que descreve: (“*Normalmente eu o uso como um autocomplete de código. Então eu estou escrevendo ali, por exemplo, eu escrevo criando uma variável chamada usuário, o copilot completa o usuário com um nome, um CPF”)*. Da mesma forma, P3 destaca: (“*Eu trabalhava muito mexendo com templates de HTML, Então era muito código repetitivo e era tudo feito à mão, assim, aí depois que o copilot chegou, isso meio que acabou com o Copilot”)*. Esse benefício específico foi observado principalmente entre os participantes que utilizam o GitHub Copilot, devido sua capacidade de oferecer sugestões que altocompletam linhas de códigos de forma contextual.

Uma outra tendência que emergiu foi o uso estratégico das IA, como o ChatGPT, como uma ferramenta que contribui para a economia de tempo dos desenvolvedores, por usar ela para evitar buscar informações complementares em outros meios, tais como, buscadores da internet, (“*No Google eu perdia muito tempo entrando nos links para verificar se tinha a informação que eu estava procurando, Já no ChatGPT você faz as perguntas e ele já retorna com uma solução”)*, P11.

Os resultados evidenciam que os desenvolvedores estão fazendo uso das IA como ferramentas de otimização das tarefas durante o processo de criação de código. A prática de utilizá-las como recursos de autocompletar, conforme destacado por diversos entrevistados, demonstrou acelerar significativamente o processo de escrita do código. Além disso, essa abordagem se estende à redução da necessidade de realizar pesquisas em motores de busca na web em busca de informações complementares, como observado por um dos entrevistados.

5.2. Desafios

Um problema identificado foi a tendência inicial dos desenvolvedores em aceitar sugestões da IA, mesmo quando essas sugestões continham erros ou não se alinhavam totalmente com os requisitos do projeto e só perceberem posteriormente, o entrevistado P6 relata essa experiência: (“*Já tive experiência ruim porque eu já subi código que eu achei que estava bom e que em produção estava inviável, entendeu? E aí não houve esse tratamento de verificar que uma coisa funciona em local, mas que em produção definitivamente não funciona”).*

Alguns desenvolvedores mencionam que, embora a IA tenha capacidade de gerar códigos, nem sempre as sugestões apresentam um nível de assertividade bom, em situações especificas é retornado sugestões imprecisas ou ruins fazendo com que eles tenham que refinar melhor sua entrada na ferramenta, conforme validado por P9: (“*Eu peço determinada coisa aí normalmente ele me traz uma coisa bem errada e aí eu vou refinando”).* O entrevistado P11 também relata: (“*É isso mesmo, na primeira vez que tu pergunta, ele não te traz a informação toda. Você tem que estar alimentando-o com mais informação, perguntando coisas mais específicas”).*

A aceitação de sugestões imprecisas pode resultar em problemas de código, como identificado por alguns entrevistados. Portanto, é importante que os desenvolvedores desempenhem um papel ativo na validação das sugestões e ajustem seu uso da IA, fazendo

perguntas mais específicas quando necessário. A colaboração com assistentes inteligentes não substitui a necessidade de intervenção humana, mas a complementa, exigindo que mantenham uma vigilância crítica e validem as sugestões antes de incorporá-las ao código. Essa adaptação inicial à IA exigiu ajustes em seus processos de trabalho e conscientização sobre os possíveis erros gerados.

6. Considerações Finais e Direções Futuras

Esta pesquisa ainda está em andamento e, até o momento, já realizamos entrevistas com 14 profissionais da área de desenvolvimento de software. Com resultados parciais, evidenciamos diversos benefícios, como refinamento de foco no qual observou-se uma melhoria na focalização de tarefas, com as assistentes inteligentes assumindo tarefas repetitivas e rotineiras, permitindo que os engenheiros se concentrassem em atividades mais complexas e estratégicas. Além disso, também percebemos que a colaboração com os assistentes inteligentes fornece um ponto de partida para os profissionais, economizando tempo na fase de planejamento. Por último, a automação e as sugestões agilizaram vários processos, otimizando a criação de código.

No entanto, também identificamos desafios, como a baixa precisão das sugestões e a necessidade de equilibrar automação e intervenção humana. Em suma, a colaboração entre engenheiros de software e assistentes inteligentes é uma tendência promissora com potencial para remodelar o desenvolvimento de software, desde que os profissionais estejam cientes dos benefícios e desafios envolvidos.

Na próxima fase da pesquisa, planejamos conduzir uma investigação mais ampla e diversificada, envolvendo uma variedade de empresas e desenvolvedores que fazem uso de diferentes sistemas inteligentes de codificação. Isso permitirá uma compreensão mais abrangente das percepções e experiências dos desenvolvedores em relação ao uso de assistentes inteligentes. Além disso, consideramos relevante explorar estratégias para mitigar os desafios identificados, desenvolvendo diretrizes e boas práticas para uma colaboração mais eficaz entre desenvolvedores e assistentes inteligentes no processo de criação de código.

Agradecimentos

Gostaríamos de agradecer aos revisores por seus valiosos comentários. Além disso, queremos agradecer ao CNPq (bolsas números 420406/2023-9 e 442779/2023-2) por financiar parcialmente esta pesquisa. Também estendemos nosso agradecimento à CAPES pelo apoio através de bolsas de estudo.

Referências

Amazon (2023). Ml-powered coding companion – amazon codewhisperer. Acessado em: 19 de setembro de 2023.

Barke, S., James, M. B., and Polikarpova, N. (2023). Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7(OOPSLA1):85–111.

Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., and Gazit, I. (2022). Taking flight with copilot: Early insights and opportunities of ai-powered pair-programming tools. *Queue*, 20(6):35–57.

Blackbox (2023). Blackbox. Acessado em: 19 de setembro de 2023.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Chang, Y., Wang, X., Wang, J., Wu, Y., Zhu, K., Chen, H., Yang, L., Yi, X., Wang, C., Wang, Y., et al. (2023). A survey on evaluation of large language models. *arXiv preprint arXiv:2307.03109*.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. (2021). Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Ernst, N. A. and Bavota, G. (2022). Ai-driven development is here: Should you worry? *IEEE Software*, 39(2):106–110.

Friedman, N. (2021). Introducing github copilot: your ai pair programmer. *URL https://github. blog/2021-06-29-introducing-github-copilot-ai-pair-programmer*.

GitHub (2022). Github copilot - your ai pair programmer. Acessado em: 19 de setembro de 2023.

OpenAI (2023). Chatgpt. Acessado em: 19 de setembro de 2023.

Perry, N., Srivastava, M., Kumar, D., and Boneh, D. (2022). Do users write more insecure code with ai assistants? *arXiv preprint arXiv:2211.03622*.

Rajlich, V. (2014). Software evolution and maintenance. In *Future of Software Engineering Proceedings*, pages 133–144.

Vaithilingam, P., Zhang, T., and Glassman, E. L. (2022). Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In *Chi conference on human factors in computing systems extended abstracts*, pages 1–7.

Weisz, J. D., Muller, M., Ross, S. I., Martinez, F., Houde, S., Agarwal, M., Talamadupula, K., and Richards, J. T. (2022). Better together? an evaluation of ai-supported code translation. In *27th International Conference on Intelligent User Interfaces*, pages 369–391.

Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., and Aftandilian, E. (2022). Productivity assessment of neural code completion. In *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, pages 21–29.